A CCSDS Software System for a Single-Chip On-Board Computer of a Small Satellite

Isabel Rutter
Tanya Vladimirova
Surrey Space Centre, University of Surrey, Guildford, Surrey, GU2 7XH, UK
Tel. +(0)1483 879278, email: ee71ir@eim.ac.uk, t.vladimirova@eim.ac.uk,

Hans Tiggeler
Surrey Satellite Technology Limited, Guildford, Surrey, GU2 7XH, UK
Tel. +(0)1483 879278, email: h.tiggeler@sstl.co.uk

Abstract. The work presented in this paper is part of a Surrey Space Centre research project that aims to reduce the size of an on-board computer to a single chip facilitating further miniaturisation of small satellites. The paper is concerned with a communication system, specifically designed to meet the needs of a single-chip on-board computer - a simplified yet reliable and automated standalone alternative software implementation of the Consultative Committee of Space Data Systems (CCSDS) protocol communicating with a standard universal asynchronous receiver-transmitter (UART) peripheral. Details of the design and implementation stages of a CCSDS package coded in the language C are given. A thin hardware layer is described which translates the asynchronous UART stream into a CCSDS compliant synchronous stream. Synthesis results targeted at Actel FPGAs are presented taking into account single event upset tolerant coding styles.

1. Introduction

The Surrey Space Centre has a long-term research programme named "ChipSat" aiming at further miniaturization of the small satellite platform. The first step of this programme is an ongoing experiment that targets at reducing the size of an On-Board Computer (OBC) to a single chip. A system-on-a-chip (SoC) that has the functionality of a small satellite OBC is being prototyped using a high-density Field Programmable Gate Array (FPGA) chip. [1] This paper presents a reliable Telecommand and Telemetry communication system based on the Consultative Committee of Space Data Systems (CCSDS) protocol specifically designed to meet the needs of a single-chip on-board computer.

Being standard space industry communication protocol CCSDS has been employed on numerous missions ranging from relatively simple low earth missions to deep space probes. Using the CCSDS standard has proven important and advantageous since it could lead to spacecraft interoperability, re-usable systems and mission cross support – not just for in-house missions but across the CCSDS space agencies members – thereby reducing costs for on-board, group and test equipment, as well as for spacecraft testing and in-orbit operations. However, it is also vulnerable to individual space agency's interpretation since users can select to implement different subsets of features and options of the CCSDS Recommendations according to their requirements. Hence, the development of this CCSDS-based system involved the need to carefully examine the influence that legacy elements may impose on the design and selection of CCSDS options.

The communication system presented in this paper comprises a simplified yet reliable software implementation of the CCSDS protocol and a thin layer hardware interface. The paper is structured as follows. Section 2 discusses design stages, structure and module interfaces of the developed CCSDS software package. Section 3 comments on the package implementation. Section 4 details the hardware layer design and Section 5 concludes the paper.

2. CCSDS Software Package Design

In this project, the main requirements were to provide a simplification of the main CCSDS TLM and TC protocol avoiding long implementation time and cost, and ensuring reliable TC acknowledgments from the spacecraft.

The development of the CCSDS software package is based on the CCSDS TLM and TC Recommendation documents, [2-6] which contain the detailed specifications of the logic required to achieve a CCSDS reliable communication system. There are optional and mandatory sections within each CCSDS Recommendation. Choosing to implement the optional sections that met the project requirements in accordance with the mandatory sections produced the simplified CCSDS TLM and TC system.

The CCSDS project was divided into two all-encompassing software sections in order to develop the CCSDS TLM and TC Command Operation Protocol (COP-1). [4] The first section developed the "CCSDS Expedited Service" [4] that implements the CCSDS TLM system and the CCSDS TC system as separate entities within four modules. The Expedited Service is concerned with the TC Type-BD frame. [4] The Type-BD frames are normally used only in exceptional operational circumstances, typically during spacecraft recovery operations. There is only one transmission for each Type-BD frame (i.e. no TC retransmission) producing an open loop communication system.

The second section developed the "CCSDS Controlled-Sequence Service" [4] that is based on an Automatic TC Request for Retransmission (ARQ) procedure with sequence-control mechanism both on the ground and on-board the spacecraft. It is also based on the necessary presence of a standard report returned in the telemetry downlink, the Command Link Control Word (CLCW). [3] These reports provide the reliable TC close loop communication system. The retransmission mechanism ensures with a high probability of success that no TC unit is lost, duplicated and is not delivered out of sequence. In this package, the COP-1 service is based on the development of a simplified version of the CCSDS Frame Acceptance and Report Mechanism (FARM) [4] and on a simplified version of the CCSDS Frame Operation Protocol (FOP). [4] Both the simplified FARM and FOP systems contain the minimal CCSDS requirements to provide a reliable CCSDS communication system.

The full package comprises six modules that provide both CCSDS TLM and TC services and are located inside two subsystems: the "CCSDS Spacecraft System" (CCSDS_SC System) – that includes the FRAM module, the CCSDS TC Rx module and the CCSDS TLM Tx module – situated inside the On-Board Computer of the spacecraft, and the "CCSDS Ground System" (CCSDS_GND System) – that includes the FOP module, the CCSDS TC Tx module and the CCSDS TLM Rx module – situated in the ground station. The diagram in Figure 1 shows the structure of the package and how all the six modules interface with each other inside the two developed systems. The modules TC Tx, TC Rx, TLM Tx and TLM Rx were developed in the first stage of this project. The modules FOP and FARM were developed in the second stage. The rest of this section provides details about the individual modules and discusses provisions for reliable telecommand transmission.

2.1. *Simplified CCSDS Expedited Service: TLM & TC Software Modules Design*

The first stage involved the software design and implementation of four modules that would be able to

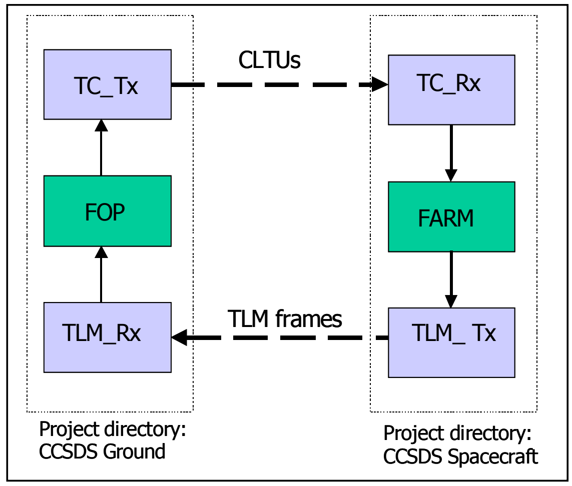

Figure 1: CCSDS Software Package Structure and Interfaces

just transmit and receive simplified CCSDS TLM and TC frames independently of each communication system. Two software modules were produced for the TLM System – the CCSDS TLM Frame Transmitter (CCSDS TLM Tx) and the CCSDS TLM Frame Receiver (CCSDS TLM Rx); and another two for the TC System - the CCSDS TC Frame Transmitter (CCSDS TC Tx) and the CCSDS TC Frame Receiver (CCSDS TC Rx).

The CCSDS TLM Tx module enables the transmission of CCSDS TLM frames [2] and its main functions are to format CCSDS TLM packets [2] of length 256 bytes; format CCSDS TLM frames of length 268 bytes; inset TLM packets into the Data Field of the TLM frames; calculate the CCSDS recommended Cyclic Redundancy Check (CRC) of the frame [2] inserted at the end of TLM frames; and finally, format the Attached Synchronized Marker (ASM) [2] – used to provide TLM frame synchronisation and is located at the start of each TLM frame.

The CCSDS TLM Rx module is implemented to decode CCSDS TLM frames received from the CCSDS TLM Tx module. Its main functions are to receive CCSDS TLM frames; subtract CCSDS TLM packets from the Data Field of the TLM frames; decode the CRC and if no error is detected, the raw TLM data is passed on to the telemetry processor for analysis and display.

The CCSDS TC Tx module is implemented to enable the transmission of CCSDS TC frames. [5] Its main functions are to format CCSDS TC packets; [5] format CCSDS TC frames; calculate the recommended CCSDS error detection check – Bose, Chaudhuri and Hocquenghem (BCH) cyclic detecting error code; [5] insert TC packets into the Data Field of TC frames; insert TC frames into Codeblocks; [5] and finally format Control Link Transfer Units (CLTU) [5] to ensure synchronisation.

The CCSDS TC Rx module is implemented to receive and decode CCSDS TC frames received from the CCSDS TC Tx module. Its main functions are to receive CLTUs; subtract the Codeblocks from the CLTUs; detect whether a transmission bit error has been identified by the BCH code - if no error detected, TC frames are subtracted from the Codeblocks and passed on to the command decoder, otherwise, the erroneous frame is discarded.

2.2. *Simplified CCSDS Controlled-Sequence Service: COP-1 Modules Design*

This design stage involved the simplification, the software design and the development of the FARM and FOP modules and their integration with the modules of the Expedited Service.

The simplified TC retransmission system under the COP-1 protocol is a closed-loop telecommanding protocol that utilises sequential ("go-back-n") retransmission technique and consists of the pair of synchronized procedures, FOP and FARM, which execute within on TC Virtual Channel [3] at the sending and receiving ends of the Transfer Layer [2]. This method ensures the retransmission of TC frames that were rejected by the spacecraft because of error.

The simplified CCSDS FARM module produces TC frame acceptances for spacecraft higher layers and develops TC report mechanisms to FOP via CLCWs inside TLM frames. Its main functions in the package are to receive error-free TC frames from the CCSDS TC Rx module; produce "Frame Validation Checks" [4] to all the incoming TC frames into the FARM system; format CLCWs depending on the FARM status; send CLCWs to the CCSDS TLM Tx module for insertion inside the corresponding TLM frames. This FARM module was simplified by only excluding the "Wait" concept from the full CCSDS FARM state machine. [4]

The simplified CCSDS FOP module processes CLCWs, received via the TLM frames from the FARM system, transmits or retransmits, as the case my be, TC frames back to the FARM system. Therefore, FOP closes the transmission loop of TC frames. Its main functions are to receive error-free TLM frames from the CCSDS TLM Rx module; subtract CLCWs from all incoming TLM frames into the FOP system; analyse the content of the CLCWs; send flags to the CCSDS TC Tx module for it to transmit a new TC frame or retransmit a previous TC frame that was not acknowledged by the FARM system. This module contains a simplified version, explained below, of the full CCSDS FOP state machine. [4]

The full CCSDS FOP protocol State Machine is very complex. It comprises six states: "Active" State (S1), "Retransmit Without Wait" State (S2), "Retransmit With Wait" State (S3), "Initialisation Without BC Frame" State (S4), "Initialisation With BC Frame" State (S5), "Initial" State (S6). The state transitions are numerous and complex providing reliable accountability of the sequence-controlled frames at the receiving-end Higher Layer through the ground station. Therefore, the full FOP protocol was simplified such that the main FOP protocol, shown in Figure 2, was preserved. This main protocol is capable of handling automatically flow control and error control.

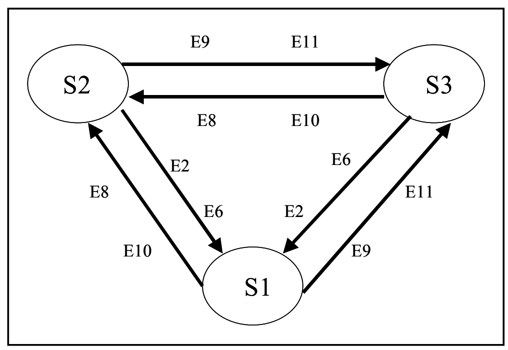

Figure 2: FOP Main State Machine Protocol

The FOP main state machine contains three states as follows. "Active" state (S1) is the normal state of the protocol machine where there are no recent errors on the link and no incidents have occurred leading to flow control problems. The "TC Retransmit without Wait" state (S2) is where the "TC Retransmit" flag is "on" in the CLCW but no other exceptional circumstances prevail and the "TC Retransmit with Wait" (S3) state is where the "TC Retransmit" and the "Wait" flags are "on" in the CLCW.

In an attempt to further simplify the FOP protocol, the "Wait" concept was not implemented as it was not necessary for the FARM system and it is optional in CCSDS. Therefore, the simplified main FOP system would still have three states but having the "Retransmit with Wait" State changed for the "Simplified Initial" State (S3*) as shown in Figure 3. This "Simplified Initial" State would ensure that transition sequences do not get forced into closed loops in the "Active" State and in the "Retransmit without Wait" State, thus, ensuring transitions between the states "Active" and "Retransmit without Wait".

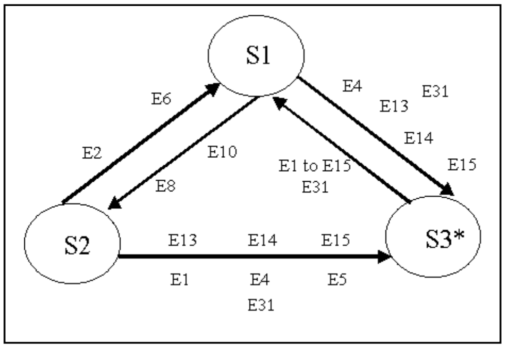

Figure 3: FOP Simplified Main State Machine Protocol

2.3. *Telecommand Transmission Reliability*

The correctness of delivered TC data units is guaranteed (within known error probabilities) by means of forward error correction applied by the Channel Service (i.e. the BCH encoding in the Codeblocks inside the CLTUs), and by the Frame Validation Check performed by the Transfer Layer in the FARM module as stated before. However, validation of the completeness, sequentially and non-duplication of the delivered data units requires a frame accounting (i.e. numbering) scheme that is implemented by the COP-1. The frame accounting method (that provides automatic retransmission of Type-AD frames [4]) is provided by COP-1 by the use of the following variables: [4] the "Receiver Frame Sequence Number" (VR) contained in the FARM module, the "Transmitter Frame Sequence Number" (VS) contained in the FOP module, the "Next Expected Frame Sequence Number" (NR) contained in the CLCW in the TLM Frame and the "Frame Sequence Number" (NS) contained in the TC Frame Header inside the CLTU. Figure 4 illustrates the relationship of the COP-1 variables.

FARM permits Type-AD frames to be accepted only is they are received bearing absolute FRAME SEQUENCE NUMBERS that are in the proper up-counting sequential order. Upon detection of the first frame sequence error, the FARM will reject all subsequent Type-AD frames which do not contain the expected FRAME SEQUENCE NUMBER (NS variable), and the FOP must go back n frames and resume sequential retransmission by repeating all unacknowledged Type-AD frames.

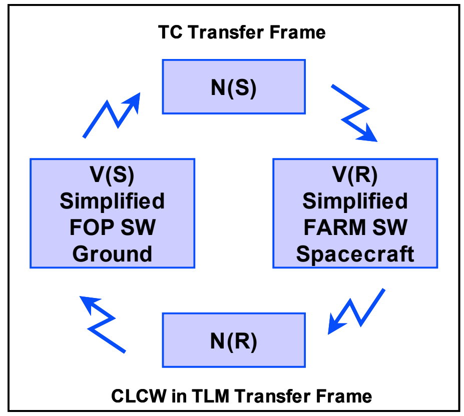

Figure 4: COP-1 Variables, Frames and TC ACK Report

3. <u>CCSDS Package Implementation</u>

This section briefly outlines the CCSDS Layering Technique and details the data structures of the TLM and TC streams - TLM Source Packet, TLM Transfer Frame, TC Source Packet and TC Transfer Frame.

3.1. *CCSDS Layering Technique*

The CCSDS TC and TLM System works under a layering technique. It was found to be a very useful tool for transforming the TC and TLM System concept into sets of simplified operational and formatting procedures. The layering approach undertaken is a seven-layer technique that groups functions logically and provides conventions for connecting functions at each layer. Layering allows a complex procedure such as the telecommanding /telemetering of command data/spacecraft data to the spacecraft/users to be decomposed into sets of peer functions residing in common architectural strata.

Within each layer, the functions exchange data according to the CCSDS protocol. Each layer draws upon a well-defined set of services provided by the layer below, and provides a similarly well-defined set of services to the layer above. As long as these service interfaces are preserved, the internal operations within a layer are unconstrained and transparent to other layers. Therefore, an entire layer within the system may be removed and replaced as dictated by a user of technological requirements (i.e. for instance, requirement of an updated system or improved version). Therefore, any software module of the package can be inserted as an individual module in a spacecraft mission and be updated or modified to convey with the specifications of such a mission without destroying the integrity of the rest of the system. Furthermore, as long as the CCSDS interface protocol is satisfied, a user can interact with the system/service at any of the component layers. Layering is therefore a powerful tool for designing structured systems, which change due to evolution of new technology or the need for different requirements. In the case of this project, change will be inevitable as the CCSDS TLM and TC system produced is the basis for additional compliant CCSDS TLM and TC components for future improvements with following projects. Figure 5 is a graphical representation of the CCSDS TLM and TC layers.

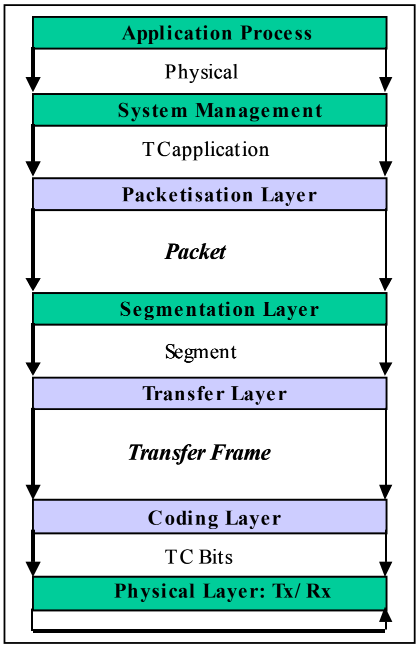

Figure 5: CCSDS Layer Technique

This project concentrated in producing the Packetisation Layer, the Transfer Layer and the Coding Layer of both the TC and TLM Systems.[2,3] The Application Process Layer and the System Management Layer [2] are not implemented as of yet. Currently, the data encapsulated inside the TLM and TC packets are idle. As the TLM and TC frames are relatively small (approx. 280 bytes) there was no need to implement the Segmentation Layer [2] either.

The TC Transfer layer [2] is the "heart" of the standard conventional TC System. It is this layer, which takes care of most of the operations required to move sets of user TC data reliably from the sending end of system to the receiving end in space. In this package, the TC and TLM layers are conveyed inside the CCSDS TC Tx, the CCSDS TC Rx, the CCSDS TLM Tx and the CCSDS TLM Rx modules.

Another CCSDS standardisation technique for data handling also implemented in the package was the conceptually simple, yet very robust, encapsulation of data within an envelope or "header". The CCSDS TLM and TC frame headers contain the identifying information needed by the TLM and TC system to provide its service while maintaining the integrity of the envelope contents.

3.2. *CCSDS TLM and TC Data Structures*

The CCSDS TLM and TC Recommendations [2-6] establish a common framework and provide a common basis for the data structures of TLM and TC streams. They specify a wide range of formatting capabilities, which facilitate a high degree of flexibility in the design of spacecraft data acquisition systems; however, compatibility with the Packet TLM and TC concept [2] was realised by only implementing a narrow subset of these capabilities. Hence, the two CCSDS defined data structures - Source Packets and Transfer Frames [2], were simplified as stated below.

3.2.1. *CCSDS TLM Source Packet*

The CCSDS TLM Source Packet Primary Header [2] contains an "Application Process Identifier" [2] used to route the packet to its destination sink. Ground station software deals automatically and immediately with the routing of telemetry packets. Such software can easily be modified to route the CCSDS packets on ground. Therefore, the "Application Process Identifier" in the Packet Header was programmed as a constant. Thus, only software to decode the CCSDS Telemetry Packet Header had to be produced in the CCSDS TLM Rx module.

The CCSDS Source Packet Header also provides the length of the packet as these can be of variable length up to a maximum of 65536 bytes plus 6 bytes of Header. Although, very long packets are permissible in CCSDS, these may present special problems in terms of data link monopolisation, source data buffering, and network accountability during transfer across the unique channel from the spacecraft to the ground and may add complexity to the ground processing and decoding. Therefore, the software was designed to decrease the possibility of CCSDS Source Packets splitting between two or more Transfer Frames. This was done by ensuring just one Telemetry Source Packet was introduced per Telemetry Frame and by fixing their length, also providing simplicity to the development of the TLM system.

Additionally, the optional Source Packet Secondary Header [2] is for time tagging. It is optional and was not implemented in order to make frames shorter (decreasing frame overhead) and aiding simplification. The total length of the TLM Source

Packet is 256 bytes (i.e. 6 bytes of Header and 250 of raw TLM data).

3.2.2. *CCSDS TLM Transfer Frame*

The CCSDS TLM Transfer Frame contains a "Transfer Frame Identifier" [2] used to identify the spacecraft that is sending the TLM data and the Virtual Channel [2] that is being used. As only one Virtual Channel was implemented (no more than two have even being used in a CCSDS communication compatible spacecraft) these bytes were made to be constants. There are also two counters - one counts the frames that go through the Master Channel [2] and one counts the frames that go through each Virtual Channel. The Master Channel and the Virtual Channel are the same, simplifying the software implementation even further.

"The First Header Pointer" [2] is the last byte of the Transfer Header, which contains information on the position of the first Source Packet within the Transfer Data Field [2]. It is programmed as a constant as the Transfer Frames are designed to have a Source Packet starting at the beginning of the Transfer Frame Data Field for reasons previously stated that also facilitate the simplification process. Additionally, the Secondary Data Field [2] is not implemented making the Transfer Frame shorter (and reducing frame overhead).

The CRC occupies the two bytes following the Data Field. It is used as protection against data corruption instead of having the Transfer Frame Reed-Solomon encoded. The decision for this was based on that the CRC is easier to implement than Reed-Solomon. Also, as the Transfer Frames are short, the CRC is sufficient as an error control, simplifying the Channel Access Data Unit (CADU) [6] to just the ASM plus the Transfer Frame.

The Operational Control Field, [6] contains the CLCW reports, and occupies the four bytes following the Transfer Frame Data Field providing the CCSDS mechanism for reporting telecommand verification.

The CCSDS Transfer Frames are by CCSDS definition of fixed length and can be any number of bytes up to a maximum of approximately 1000 bytes. For this package the TLM frame length was chosen to be 268 bytes (4 bytes of ASM plus 6 bytes of Header plus 256 of Data Field plus 2 bytes of CRC).

3.2.3. *CCSDS TC Source Packet*

The TC Source Packet Header [6] is 6 bytes long and is divided into three sections. The first section is called "Packet Identifier" [6] and it contains the version number and the type of the packet, and also a flag to indicate to the CCSDS TC Rx module whether the packet contains a secondary header or not. For the purposes of simplification and in order to keep the packet overhead small, the secondary header was not included in the CCSDS TC packet. The implementation of the secondary header is optional, and therefore, the TC packet implemented still conveys with the CCSDS protocol. The "Packet Identifier" also contains the "application ID" [6] of the packet. The "application ID" informs the TC decoder what application does the data in the packet data fields should go to. As the TC system is implemented in a simulation environment, the data introduced into the CCSDS TC Source Packets was idle data. Consequently, the "application ID" was programmed to be a constant. Additional C code to modify this section can be added very easily and without affecting other parts of the CCSDS TC Tx module or the other CCSDS modules implemented.

The second section of the TC Source Packet is called the "Packet Sequence Control" [6] and it is divided into two parts. The first one is a flag composed of 2 bits indicating a stand-alone packet that does not form part of a group of packets. The second part is the "Source Sequence Count" [6] that enables the CCSDS TC Rx module to verify whether the packet received is the expected one or not.

The third part is the "Packet Length" [6]. These two bytes inform the CCSDS TC Rx module of the number of real TC data bytes (i.e. data inside the TC Source Packet) that the TC source packet is carrying. For this package, the entire length of the TC Source Packet is 275 bytes (6 bytes of Packet Header plus 269 bytes of TC Packet Data Field).

3.2.4. *CCSDS TC Transfer Frame*

The TC Frame header (Figure 6) is five bytes long. They include the version number (i.e. to indicate the origin of the TC Frame) followed by two flags and the spacecraft ID (to indicate if the TC Frame received was intended for the spacecraft that received the Frame). The two flags are very important. The "Bypass flag" [6] provides a high-level priority indication in the CCSDS TC Rx module. This means that the TC Frame that has the "Bypass flag" activated has a higher priority of being processed over other TC frames that have not got this flag activated. The "Control Command Flag" [6] indicates that the data contained in the TC Frame Data Field is a command and not a TC Source Packet.

The following two bytes contain the "virtual channel ID" [6], programmed as a constant for reasons already stated above, and the "frame length". The "frame length" has the same function as it has in the TC

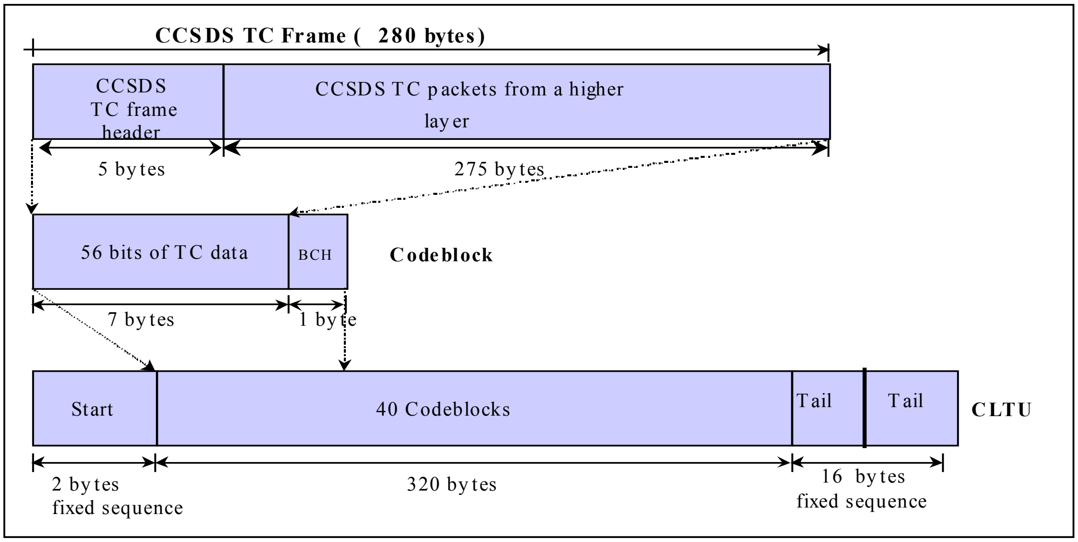

Figure 6. **CCSDS TC Data Structure**

Source Packet, but this time it indicates the length of the entire frame rather than just the length of the data field.

The last byte of the TC Frame Header is the variable N(R). The entire length of the TC Frame is 280 bytes (5 bytes of Frame Header plus 275 bytes of TC Frame Data Field).

The data inside the frame data field is also implemented so as to be idle for this project. Attached to the end of each TC Frame can be a CRC. Its implementation is optional under the CCSDS protocol and for simplicity reasons it was not produced. Once the TC frame is developed, it is divided into Codeblocks and inserted into CLTUs as shown in Figure 6. The CLTU is 338 bytes long in total.

4. Hardware Layer Design

In order to simplify the processor interface a relatively simple thin hardware layer was developed. This hardware layer allows any processing element with a UART to communicate with the synchronous CCSDS data stream. The layer also significantly reduces the processing overhead by providing octet and frame synchronization on the uplink. The diagram in Figure 7 shows the UART to CCSDS interface.

The uplink interface consists of a frame synchronizer, which scans for the 16-bit CLTU start sequence. Once the sequence has been detected all following octets are encapsulated with a start and stop bit and transmitted to the On-Board Computer. The layer continues the encapsulation process until the UART Data Carrier Detect (DCD) is negated (TC Channel Service State S1, no bit modulation). The application layer can force a frame resynchronisation (TC Channel service state S2, start sequence search) by negating the Data Set Ready signal (DSR).

The downlink interface removes the start and stop bit from the received UART character and transmits it to the synchronous output. A small finite state machine is added to provide flow control using the UART Request To Send (RTS) and Clear To Send (CTS) handshake signals.

The hardware layer has been coded in VHDL and synthesised to an ACTEL FPGA. Table 1 presents the synthesis results for the smallest Actel 5 4SX08 FPGA and the bigger 54SX16 FPGA. As shown in Table 1 the converter requires just 30% of the 54SX08 FPGA. Single Event Upset (SEU) tolerant VHDL coding techniques[7,8] have been applied which increased the cell count to 44% for the C-cell only implementation and to 58% when using full CC-cell Triple Modular Redundancy (TMR). For the latter a larger 54SX16 FPGA is required.

Table 1: Synthesis Results (Synplicity 6.0, Actel Designer R2-1999)

SEU Coding Technique	**54SX08** Total Cells (768)	**54SX16** Total Cells (1452)
None	**230 (30%)**	221 (15%)
CC	343 (44%)	343 (23%)
TMR	*Doesn't fit*	844 (58%)
TMR_CC	*Doesn't fit*	846 (58%)

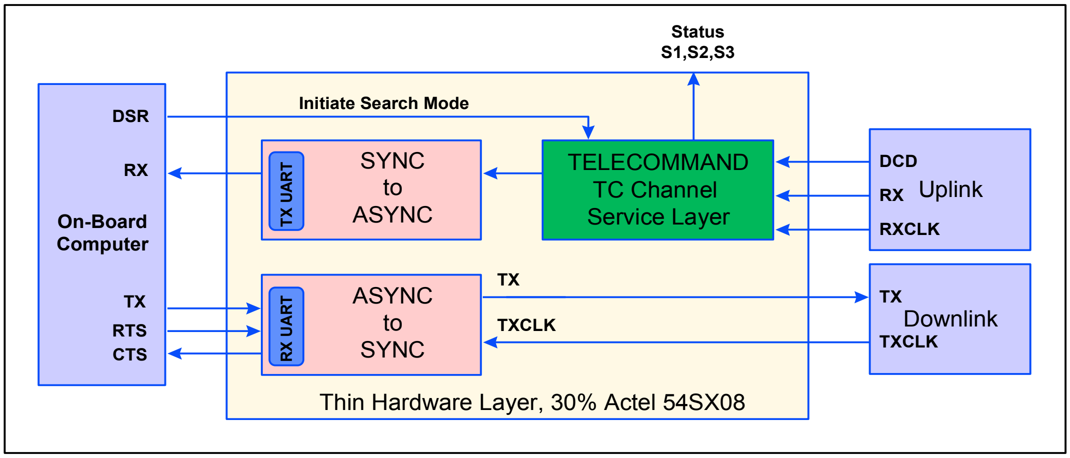

Figure 7. Structure of the Hardware Layer

Adding the BCH Codec to the thin hardware layer can further reduce the processing requirements. This is currently in development. It must be noted that the presented hardware layer and associated OBC can not replace the standard hardwired telecommand system without additional fault tolerant techniques. The OBC or processing element is used to verify, decode, distribute or execute the received telecommands. Additional circuitry is required to re-initialise the OBC in case of a system crash. The thin hardware layer can be enhanced with a hardwired BD-frame decoder to perform this task.

5. Conclusions

At present, the complete CCSDS TLM and TC implementation is still very complex for low-cost small satellites and hardware implementations are expensive. In this paper we present a cost effective and flexible communication system that consists of a simplified, yet reliable and automated, standalone alternative software implementation of the CCSDS protocol and a thin layer hardware interface. The software CCSDS implementation imposes minimal memory footprint and performance requirements on the On-Board Computer. The thin hardware layer not only translates between the synchronous CCSDS stream and asynchronous UART stream but also handles frame and octet synchronisation on the uplink and CRC and frame generation on the downlink. The hardware layer allows any processing element with a UART to communicate with the synchronous CCSDS data stream without being purposely modified to do that. This communication system, specifically designed to meet the needs of a single-chip on-board computer, will facilitate further miniaturisation of small satellites

This project did not intend to exploit the CCSDS features or provide maximum efficiency but provide a simplified yet reliable CCSDS TLM and TC System. Development of more sophisticated CCSDS TLM and TC system may follow this project. The CCSDS software package features a modular structure which can be very advantageous. It will facilitate easy expansions of functionality in the future to include optional CCSDS features without affecting the system as a whole as well as reuse of modules on other missions. Reuse of systems for missions that adhere to the same data system standard provides the potential for large cost and reduced operations risks whereby flight operations teams can re-apply the experience gained during operations on one mission to another mission.

The CCSDS Recommendations include a variety of features and options to choose from, which makes them vulnerable to individual interpretation. Two different organisations can take the same Recommendations and develop systems that will not interpolate completely. Users that embark on CCSDS development should select a subset of features and options that best meets their requirements, but also should take into account the CCSDS features required to work with other organisation's systems.

Acknowledgements

The authors would like to acknowledge the help and advice of Rob Myatt and Chris Jackson from SSTL on the CCSDS protocol and its simplification.

References

1. H.Tiggeler, T.Vladimirova, D.Zheng, J.Gaisler. "Experiences Designing a System-on-a-chip for Small Satellite Data Processing and Control" - Proceedings of International Conference on Military and Aerospace

Applications of Programmable Devices and Technologies (MAPLD'2000), P-20, September 2000, Laurel, Maryland, US, NASA.

2. "Packet Telemetry Recommendation for Space Data Systems Standards", CCSDS 102.0-B-4. Blue Book. Issue 4. Washington, D.C.: CCSDS, November 1995

3. "Recommendation for Telemetry Channel Coding" [6], CCSDS 101.0-4. Blue Book. Issue 4. Washington, D.C.: CCSDS, May 1999

4. "Telecommand Part 2.1. Recommendation for Telecommand: Command Operation Procedures", CCSDS 202.1-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, October 1991.

5. "Telecommand Part 2. Recommendation for Telecommand: Data Routing Service", CCSDS 202.0-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, November 1992.

6. "Telecommand Part 1. Recommendation for Telecommand: Channel Service", CCSDS 201.0-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, November 1995.

7. "Design Techniques for Radiation Hardened FPGAs", Actel, Application Note, September 1997, http://www.actel.com/appnotes/5192642.pdf

8. "Using Synplify to Design in Actel Radiation-Hardened FPGAs", Actel, Application Note, May 2000, http://www.actel.com/appnotes/SynplifyRH.pdf

Isabel Rutter is a recent BEng graduate in Electronic and Electrical Engineering from the School of Electronics, Computing and Mathematics of the University of Surrey and an Associate Member of the Institute of Electronic and Electrical Engineers (IEEE). She has experience in the CCSDS communication protocol and was awarded the 'IEEE Best Undergraduate Project in the Telecommunications Area' for her CCSDS project.

Tanya Vladimirova (MEng, MSc, PhD, CEng, MIEE, MIEEE) is a Senior Lecturer at the School of Electronics, Computing and Mathematics of the University of Surrey and a member of the Surrey Space Centre. She teaches ASIC Design and VLSI Systems and Computer Architecture. Her research interests are in the area of high-performance VLSI systems, system-on-a-chip and computational intelligence.

Hans Tiggeler is a Senior Engineer and a leader of the On-Board Data Handling team of the Surrey Satellite Technology Limited. He has extensive experience in on-board computer design as well as FPGA design. His research interests are in spacecraft electronics, advanced computer architecture, system-on-a-chip, ASIC and FPGA design.